Learning: Baum-welch Algorithm

Baum-welch Algorithm is a method to tune the parameters of the HMM based on estimations that already done from a given data $\boldsymbol{\lambda}\ (P, E, \pi)$. By iterating using "Expected Maximum" algorithm to reach a local (may be a global) optimum estimation of $\boldsymbol{\lambda}$ constructing a new model $\widehat{\boldsymbol{\lambda}}\ (\widehat{P}, \widehat{E}, \widehat{\pi})$.

The estimation of the model parameters is calculated as following:

1. Initial distribution ($\widehat{\pi}$): is the expected frequency of being in (i) at (t=1). Which is calculating from a parameter called gamma (γ_t^i).
   - γ_t^i : refer to the probability of being in (i) at time=t.

$$\gamma_t^i = \frac{\alpha_t^i \beta_t^i}{\sum_{i=1}^{N} \alpha_t^i \beta_t^i} = \sum_{j=1}^{N} \varepsilon_t^{ij} \qquad \hat{\pi} = \gamma_1^i$$

2. Transition matrix ($\widehat{P}$):
   Each transition probability Pij is estimated as "the expected number of transitions from i to j" and is calculated from a parameter (ξ_t^{ij}) over the "expected number of transitions from i to any other state" .

$$\varepsilon_{ij}(t) = \frac{\alpha_i^t P_{ij} e_j(O_{t+1}) \beta_{t+1}^j}{\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i^t P_{ij} e_j(O_{t+1}) \beta_{t+1}^j} \qquad P_{ij} = \frac{\sum_{t=1}^{T-1} \varepsilon_t^{ij}}{\sum_{t=1}^{T-1} \gamma_t^i}$$

3. Emission matrix ($\widehat{E}$):
   Each Emotion $e_j(K)$ is the "expected number of times being in j subject to observation k" over "probability of being in j"

$$e_j(k) = \frac{\sum_{t=1}^{T} \gamma_j^t, s.t._{O_t=k}}{\sum_{t=1}^{T} \gamma_j^t}$$

Algorithm steps with implementation in R

1. Given initial model $\boldsymbol{\lambda}\ (P, E, \pi)$, α *and* β, sequence of observations and number of iterations.

```
Baum_welch <- function(P,E,pi,alpha,beta,seq,n_iter) {
  itrations = n_iter
  T = length(seq)
  n = length(S)
  n_obs = length(O)
  expi <- array(0,dim = c(n,n,T-1))

  for (i in 1:itrations){}
```

First: All needed values for later calculations are defined

T: total time

n: number of hidden states

n_obs: number of observable events

expi: multidimensional list of the probability to transfer from state i to hidden state j at time t.

Second: A loop of the algorithm for a specific number of iterations.

2. The approach of the baum-welch usually starts with forward and backward evaluations of the initial model α_t^i, β_t^i as T*n matrices (have been calculated in the evaluation phase).
3. Then get ξ_t^{ij} for all states i and j at any time t in a multi-dimensional list where the first dimension is the each time point t and contain a n*n matrix of transitions.

```
for (i in 1:itrations){

  # exp(ij) matrix
  for(t in 1:T-1){
    sum_ij_t <- ((alpha[t,] %*% P) * E[,seq[t+1]]) %*% matrix(beta[t+1,])

    for(s in 1:n){
      exp_ij_t = alpha[t,s] * P[s,] * E[,seq[t+1]] * beta[t+1,]
      expi[s,,t]=exp_ij_t/as.vector(sum_ij_t)
    }
  }
```

Explanation:

1. For each time step t the denominator (sum_ij_t) is calculated
2. In the same time t for each state (i) the numerator (exp_ij_t/as.vector(sum_ij_t)) for all states (j)

4. Then get γ_t^i as a n*T matrix for all states at any time t

```
# gamma matrix
gamma = apply(expi, c(1, 3), sum)
gamma = cbind(gamma, colSums(expi[, , T-1]))
```

Explanation:

1. Calculate sum of ξ^{ij} at each time t to get the probability of being in i at t.
2. As the ξ^{ij} refers to transition with T-1 steps, To calculate the being in i at T (last column in i) it will be the sum of transition to each state i from any state at time T-1.

5. Get the estimated model parameter using the previous calculations

5.1. Estimated initial dist

```
# estimated lambda
# 1. estimated initial dist: exp freq. of being in (i) at t=1
pi <- gamma[,1]
```

5.2. Estimated transition matrix

```
# 2. estimated transition matrix
expi_t <- rowSums(expi, dims = 2) # exp trans from (i) to (j)
for(r in 1:n){
  P[r,] <- expi_t[r,] / as.vector(sum(gamma[r,1:T-1]))
}
```

Explanation:

1. The (expi_t) is the summation of probabilities to transfer from state (i) to(j) at each time step t
2. Each row (i) is divided by the probability to transfer from (i) at any time t

5.3. Estimated emission matrix

```
# 3. estimated emission matrix
colnames(gamma) <- seq
rownames(gamma) <- S
for (obs in seq){
  E[,obs] <- rowSums(as.matrix(gamma[, which(seq==obs)])) # exp be in (j)|O=k
}
for (e in 1:n){
  E[e,] <- E[e,] / as.vector(sum(gamma[e,]))
}
```

Explanation:

1. The summation of being in state (j) for each observation in the sequence (E_hat[,obs])
2. Then multiplied by probability of being in state (j) (E_hat[i,] / as.vector(sum(gamma[i,])))

6. After the given number of iteration the algorithm will return the new estimated model $\hat{\lambda}\,(\hat{P}, \hat{E}, \hat{\pi})$. (the ξ_t^{ij} and γ_t^i are additional)

```
BW_output <- list("Expected ij" = expi,"Gamma matrix" = gamma,
                  "initial dist" = pi,"Transition matrix" = P,"Emition matrix" =E)
  return(BW_output)
}
```

```
$`initial dist`
[1] 0.0056316127 0.0008038818 0.9935645055

$`Transition matrix`
              clear         rain         warm
clear 1.932379e-01 8.052490e-01 1.513046e-03
rain  2.995904e-01 7.004096e-01 1.312186e-12
warm  1.546207e-17 7.272794e-19 1.000000e+00

$`Emition matrix`
              low       med       high
clear 0.37123476 0.5364457 0.09231958
rain  0.64168304 0.3387445 0.01957246
warm  0.02826021 0.3976118 0.57412795
```